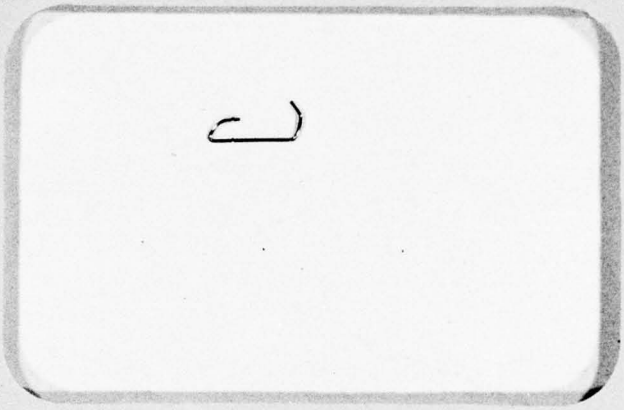
AD-A038 209 TECHNOLOGY SERVICE CORP SANTA MONICA CALIF F/G 12/2
TOPICS IN THE ANALYSIS AND OPTIMIZATION OF COMPLEX SYSTEMS.(U)
FEB 77 W S MEISEL, L BREIMAN F44620-76-C-0069
UNCLASSIFIED AFOSR-TR-77-0376 NL

| OF |
AD
A038209

END

DATE
FILMED
5-77

Technology Service Corporation



Technology Service Corporation

2811 WILSHIRE BOULEVARD • SANTA MONICA, CALIFORNIA 90403 • PH. (213) 829-7411

TOPICS IN THE ANALYSIS AND OPTIMIZATION OF COMPLEX SYSTEMS

Final Report

Air Force Office of Scientific Research
Contract No. F44620-76-C-0069

Principal Investigators:

William S. Meisel
Leo Breiman

February 28, 1977

TOPICS IN THE ANALYSIS AND OPTIMIZATION OF COMPLEX SYSTEMS

This is a final report on research carried out at Technology Service Corporation under Contract No. A151 F44620-76-C-0069. The length of the contract was one year. The purpose was to carry out research leading to more effective methods of analyzing high dimensional data sets. The motivation for the research came from our growing realization, in handling many high dimensional data sets gathered in many different fields, that classical methods were often inappropriate and when used, lead to misleading results.

Our main efforts over the year were in three areas:

First: Extensive revisions of our work on variable kernel density estimates, leading to a paper accepted for publication by Technometrics, to appear in their May 1977 issue.

The following general comments by the final referee of the paper we find particularly interesting as we believe they signify a beginning acceptance by the statistical community of the need for new methods to deal with current problems.

REFEREE'S REPORT

GENERAL COMMENTS

This is an interesting paper, describing innovative and useful research on estimating multivariate density functions in a computationally feasible way. The mathematical presentation is reasonably clear, and the simulation examples add a vital element of numerical insight.

A copy of the final version of this paper is included as an appendix.

Secondly: Under a previous AFOSR contract, a novel goodness-of-fit test devised by Leo Breiman had been tested under numerous simulations. These led to the conjecture that the test was asymptotically distribution free. The first version of the paper was submitted to JASA and rejected on the grounds that the main conjecture, while made plausible by the simulations, was not proven. Towards the end of the contract period, Leo Breiman working in collaboration with Professor Peter Bickel, chairman of the Statistics Department at U. C. Berkeley, managed to find a proof that established the asymptotically distribution free property of the test. This is currently being written up for submission to the Annals of Statistics. We consider this to be a highly significant break through. It provides the first computationally feasible consistent and asymptotically distribution free goodness-of-fit test for dimensions higher than one.

Third: The most exciting research for us over the past year has been the development of free-structured classification methods and the growing realization of their potential in approaching a large variety of problems that were untouchable by classical methods. The progress in this work was reported on by Leo Breiman at a joint U. C. Berkeley-Stanford Statistical Colloquim in October 1976 and generated a good deal of interest. Numerous requests for written descriptions of the work have been received. However, to date, we consider our work in an exciting but exploratory phase. We are looking forward to further developments and applications. A write up of our progress to date and directions we want to explore in the future are contained in an appendix.

In summary, we feel that a significant amount of innovative and useful research has been accomplished over the contractual period. Rather than launch into a long discussion of why the results are important, we prefer to let the contents speak for themselves.

APPENDIX A

Variable Kernel Estimates of Multivariate Densities and Their Calibration

Leo Breiman
William Meisel
Edward Purcell

1. Introduction and Summary

Given points $\underline{x}_1,\ldots,\underline{x}_n$ selected independently from some unknown underlying density $f(\underline{x})$ in M-dimensional Euclidean space, the problem is to estimate $f(\underline{x})$. To date, the most effective general method is the Parzen approach: select a kernel function $k(\underline{x})\geq 0$, with

$$\int k(\underline{x})d\underline{x} = 1 \tag{1}$$

Usually $k(\underline{x})$ satisfies some additional conditions; unimodality with peak at $\underline{x}=0$, smoothness, symmetry, finite first and second moments, etc. In fact, in actual practice, the most frequently used kernel is a Gaussian density.

Having selected a kernel, then the estimate is given as

$$\hat{f}(\underline{x}) = \frac{1}{n}\sum_{1}^{n}\frac{1}{\sigma^M}K\left(\frac{\underline{x}-\underline{x}_j}{\sigma}\right)$$

As n increases the shape factor σ can be decreased giving greater resolution for larger sample sizes. The asymptotic mean square consistency of these estimates is well known [1], and under smoothness conditions on $f(\underline{x})$ asymptotic rates of convergence of the mean squared error can be derived.

However, in terms of practicalities, the situation is far from satisfactory.

First: It is obvious that a Parzen method of estimation cannot respond appropriately to variations in the magnitude of $f(\underline{x})$. For instance, if there is a region of low $f(\underline{x})$ containing, say, only one sample point $\underline{x}_k$, then the estimate will have a peak at $\underline{x} = \underline{x}_k$ and be too low over the rest of the region. In regions where $f(\underline{x})$ is large, the sample points are more densely packed together, and the Parzen estimate will tend to spread out the high density region. Thus, the problem is that the peakedness of the kernel is not data-responsive.

Second: None of the asymptotic results give any generally helpful leads on how the shape factor σ should be selected to give the "best" estimate of unknown density. The computed rates of convergence depend critically on $f(\underline{x})$ and its derivatives. Even if one tried to vary σ and got a number of different estimates, the question remains: which one is "best"?

In this paper, solutions are proposed to both of these problems.

First: To make the sharpness of the kernel data-responsive, we use the class of estimates

$$\hat{f}(\underline{x}) = \frac{1}{n} \sum_{j=1}^{n} (\alpha_k \, d_{j,k})^{-M} \, K\left(\frac{\underline{x}-\underline{x}_j}{\alpha_k \, d_{j,k}}\right)$$

where $d_{j,k}$ is the distance from the point $\underline{x}_j$ to its k^{th} nearest neighbor, and α_k is a constant multiplicative factor. The intuitive concept is

clear: In low density regions, $d_{j,k}$ will be large and the kernel will be spread out. In high density regions, the converse will occur.

Second: To select optimizing values of k and α_k, a goodness-of-fit statistic $\hat{S}$ for multivariate densities proposed in [2] is used in a procedure that searches for the variable kernel parameters that minimize $\hat{S}$.

The analytics of the variable kernel estimates situation are a bit difficult to handle, although asymptotic consistency for appropriate kernels is easily proved under the condition $k/n \to 0$. To get a feeling for the finite sample situation and also to get some measure of assurance that our proposed "solutions" had some value, we ran some extensive simulations on two underlying data bases; the first was 400 points selected from a bivariate normal distribution, the second was the bimodal distribution consisting of a superposition of two bivariate normals, 3/4 of the bivariate normal used in generating the first data set plus 1/4 of a normal with a much sharper peak.

Three measures of error were computed: define the sample mean and variance of $f(\underline{x})$ by

$$\mu_f = \frac{1}{n} \sum_{1}^{n} f(\underline{x}_j)$$

$$\sigma_f^2 = \frac{1}{n} \sum_{1}^{n} (f(\underline{x}_j) - \hat{\mu}_f)^2$$

The error measures were:

I. Percent of Variance Not Explained (PVNE)

$$PVNE = \frac{1}{\sigma_f^2} \cdot \frac{1}{n} \sum_{1}^{n} (f(\underline{x}_j) - \hat{f}(\underline{x}_j))^2 \times 100$$

II. Mean Absolute Error, Percent (MAE)

$$MAE = \frac{1}{n\mu_f} \sum_{1}^{n} |f(\underline{x}_j) - \hat{f}(\underline{x}_j)| \times 100$$

III. Mean Percent Error (MPE)

$$MPE = \frac{1}{n} \sum_{1}^{n} \frac{|f(\underline{x}_j) - \hat{f}(\underline{x}_j)|}{f(\underline{x}_j)} \times 100$$

A large number of runs were carried out with the two data bases to

(A) Find the best Parzen estimator and the best variable kernel estimator, using a symmetric Gaussian kernel (naturally the "best" values of the kernel parameters depend on what measure of error is used).

(B) Compare the performances of the two types of estimators.

(C) To see whether the proposed search procedure could accurately locate the "best fitting" estimates.

Our conclusions are:

i. In all cases the best variable kernel estimate was superior to the best Parzen estimate. The best Parzen estimator had in both data sets about twice as much mean percent error (MPE) and percent of variance not explained (PVNE), and about 50% more mean absolute error than the best variable kernel estimator.

ii. The $\hat{S}$ minimization search procedure was successful in locating the region of parameter values where the variable kernel estimates gave approximately best fits to the actual density.

The best values of σ for the Parzen estimates depended on which measure of error was used much more than the variable kernel method and hence would be much more difficult to use in practice (when f is unknown). The S minimization procedure applied to the Parzen estimates produced values of σ that were larger than most of the "best" values and could not be called successful in this context.

During the course of the study, a number of interesting and useful properties of variable kernel densities were uncovered. First of all, the nearest neighbor distances that produced the best fits were surprisingly large, ranging from 40 in data set II to 100 in data set I, (actually the fit was still improving at k=100). But good fits can be produced over a very wide range of values of k, as long as α_k satisfies the approximate relation

$$\frac{\alpha_k(\overline{d_k})^2}{\sigma(d_k)} = \text{constant}$$

where $\overline{d_k}$ is the mean of the k^{th} nearest neighbor distances and $\sigma(d_k)$ is their standard deviation. Our tentative conclusion is therefore that actually one needs to find only the single parameter value $[\alpha_k(\overline{d_k})^2/\sigma(d_k)]$ to calibrate the variable kernel estimates. In our simulation this constant was usually about 3-4 times larger than the best values of σ for the corresponding Parzen estimate.

The conclusion that the mean percent error is markedly different between the two types of estimators has important implications for classification. The method giving the minimum expected misclassification probability

is based on comparing the densities of the different classes. One common and effective method of getting "good" classification boundaries has been to estimate the class densities using a set of points that have already been classified, and compare the estimates to make the classification decision. Therefore, if this is the intended application, then the mean percent error is the appropriate error measure since the tails of the distribution are important, and in this perspective the variable kernel estimates are decidely superior to the Parzen estimates.

An important consideration is the variability of the underlying density. If it is more or less uniformly smooth (as in the first data base), the adaptive capability of the variable kernel method does not help us much as in situations where the density is more variable, i.e., has a number of peaks of different sharpness (as in the second data base).

There is a large body of published literature regarding density estimation and a number of good surveys are available [3], [4], [5]. The k^{th} nearest neighbor estimator [6] is the only method that is adaptive to local sample density. If the distance from a point $\underline{x}$ to its k^{th} nearest neighbor is d, then the estimate is defined as

$$\hat{f}(x) = \frac{k/n}{V(d)}$$

where n is the total number of samples, and V(d) is the volume of the M-dimensional sphere of radius d. The drawback to this type of estimate is that it is discontinuous and that it does not satisfy (1). The variable kernel approach offers a combination of the desirable smoothness properties of the Parzen-type estimators with the data-adaptive character of the k-nearest neighbor approach.

Furthermore, the variable kernel method carries very little computational penalty. The distance from a given point to the k^{th} nearest point is computed only once and stored for all the calibration runs. An algorithm constructed by Friedman, et al. [7] reduces the finding of all k^{th} nearest neighbors to n log n time instead of n^2.

In Section 2 we will describe the simulations in more detail and give some tabular and graphical summaries of the results.

Section 3 will give a brief description of the goodness-of-fit statistic and give tabular and graphical summaries of its performance.

In Section 4, the behavior of the estimates will be summarized, the selection of k and α related to the interpoint distance distribution, and a description given of some early and unsuccessful efforts at variable kernel estimates.

The variable kernel method has been described in short course notes on pattern recognition prepared by one of the authors and dating back to 1973. The work in this present study has been reported on in the Conference on the Interface Between Computer Science and Statistics on February 14, 1975 [8]. In June, 1975 we learned that T. J. Wagner has submitted a paper [9] to the IEEE Trans. Information Theory which is also concerned with the variable kernel estimates. Since his paper is reportedly concerned with conditions for asymptotic consistency, particularly in one dimension, there does not seem to be any overlap.

2. The Simulation and Its Results

The two data sets mentioned in the introduction were generated as follows:

Set I: 400 points selected independently from the density f, a bivariate normal with mean $\underline{m} = (0,0)$ and unit covariance matrix.

Set II: 400 points selected independently from the density g, where

$$g = .75f + .25f_1$$

where f is as above, and f_1 is normal with parameters

$$\underline{m} = (3,3), \quad \Gamma = \begin{pmatrix} 1/9 & 0 \\ 0 & 1/9 \end{pmatrix},$$

where Γ is the covariance matrix.

The kernel for both types of estimators was a zero mean bivariate normal density with unit covariance matrix.

Figure 1 is a graph of the three error measures in data set I as a function of the shape parameter σ of the Parzen estimators.

Figure 2 is a graph of the three error measures for data set I, where we selected $k = 100$ and varied the multiplicative parameter α.

Figures 3 and 4 are the analogous graphs for data set II, where we have used $k = 40$ in the variable kernel graph.

In all cases, we ran the simulations until the minimal values of the three measures of error were found, both for the Parzen and variable kernel estimators. For the variable kernel estimators we ran the

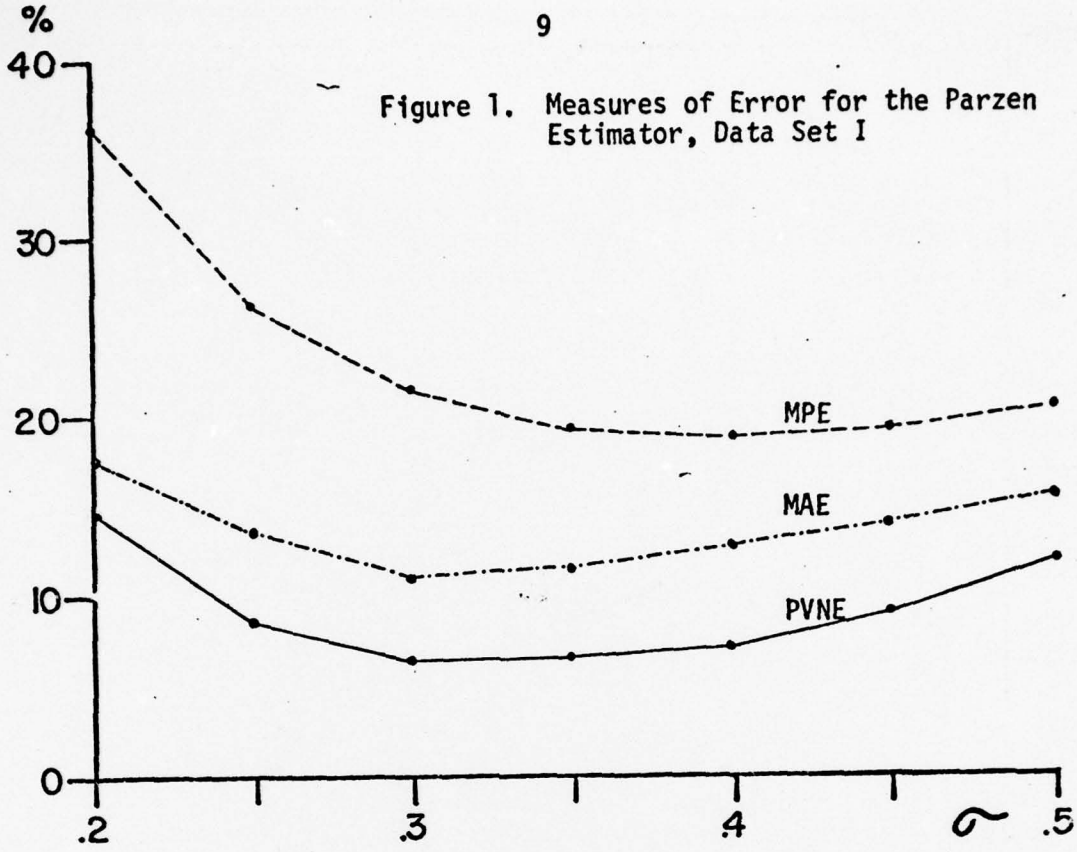


Figure 1. Measures of Error for the Parzen Estimator, Data Set I

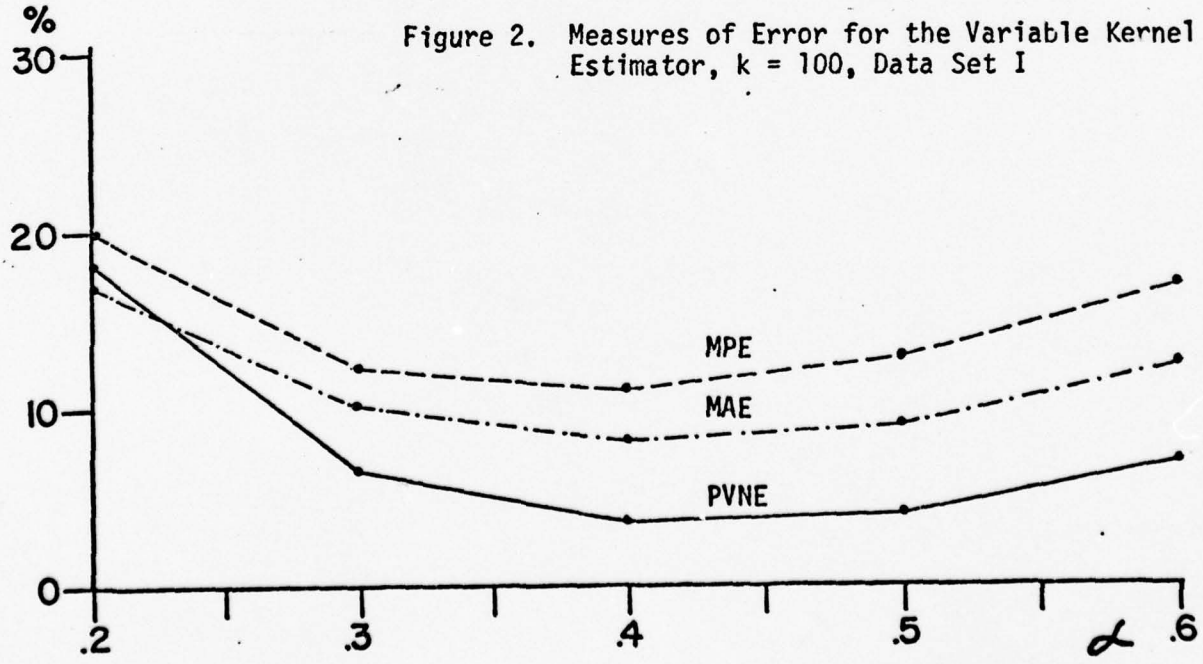


Figure 2. Measures of Error for the Variable Kernel Estimator, k = 100, Data Set I

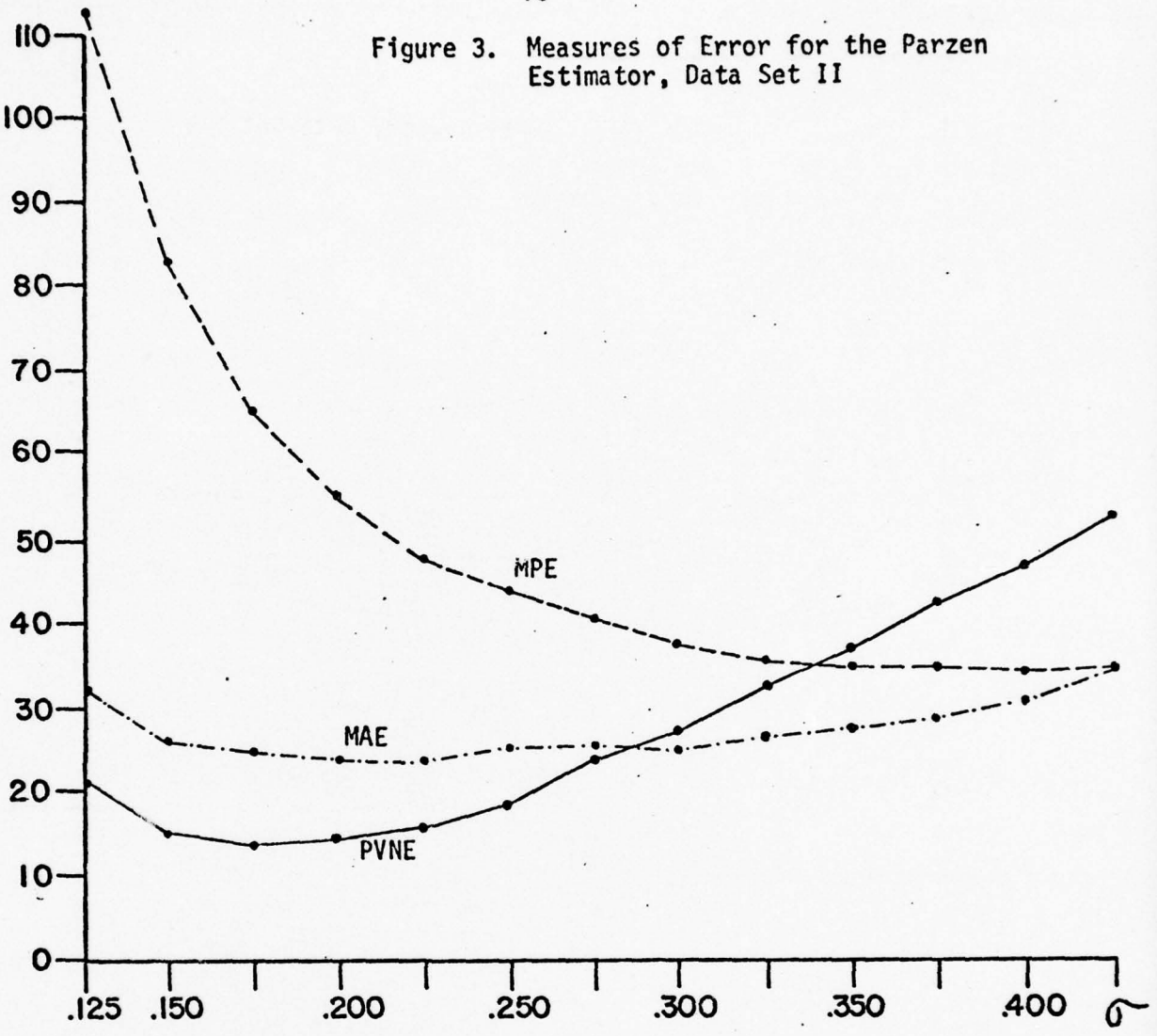


Figure 3. Measures of Error for the Parzen Estimator, Data Set II

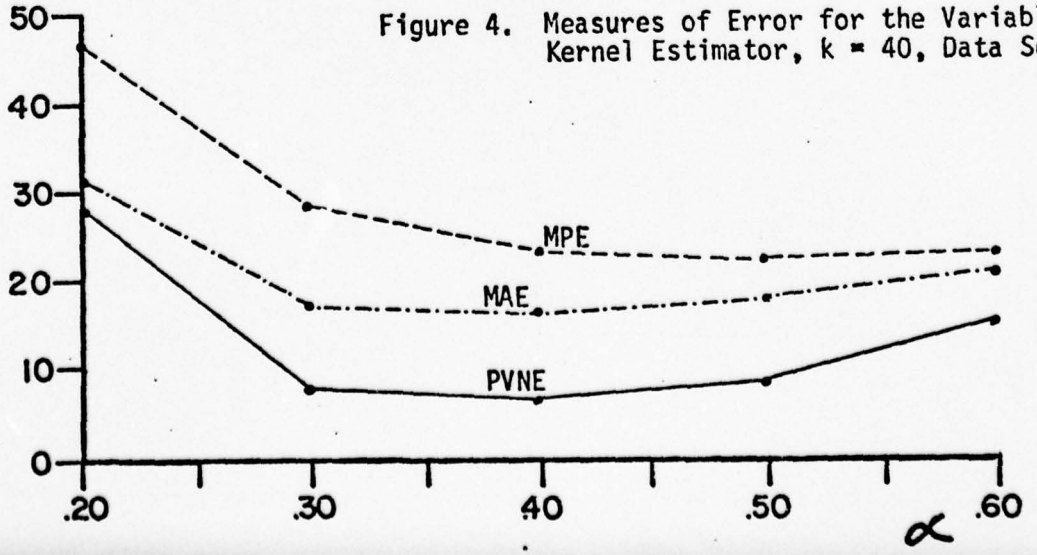


Figure 4. Measures of Error for the Variable Kernel Estimator, k = 40, Data Set II

simulations for k=10, 20, 30, 40, 50, and 60 in both data sets, and for k=70, 80, 90, 100 in data set I. Table 1 below summarizes the comparison between the methods,

To illustrate the resulting fits more visually, we plotted 3 dimensional graphs of the best estimates. For data set I, we used $\sigma = .35$ for the Parzen estimator and $k = 60$, $\alpha = .6$ for the variable kernel estimator. In data set 2, the choice of an "optimal" σ was more problematical. We settled on .275 as a reasonable compromise. For the variable kernel we took $K = 40$, $\alpha = .5$. The results are shown in figures 5, 6, 7, and 8 (see end).

	Minimum Mean Percent Error	Minimum Percent of Variance Not Explained	Minimum Mean Absolute Error, Percent
Parzen, Data Set I	19.0	6.2	11.6
Variable Kernel, Data Set I	10.8	3.0	8.0
Parzen, Data Set II	34.7	13.4	24.2
Variable Kernel, Data Set II	22.5	6.2	16.5

Table 1

Fortunately, the variable kernel results were surprisingly unsensitive to the choice of k. Table 2 below gives the minimum values of the measures of error for the different values of k. Note that in both examples, values of k over almost the entire range give quite comparable error measurements.

As k varies the fit behaves slightly different for the two data sets. For the smooth density of the first example, the error measures are still

decreasing at k=100 and we would probably have gotten slightly better results by going on to larger k. For the second density the error measures decrease up to k=40 and then increase at k=50 and 60, (except for the MPE).

Data Set I

k =	10	20	30	40	50	60	70	80	90	100
Minimum Mean Percent Error	12.9	12.8	12.2	12.1	11.7	11.6	11.4	11.3	10.9	10.8
Minimum Percent of Variance Not Explained	9.3	6.8	6.3	5.9	5.1	4.8	4.6	4.1	4.0	3.6
Minimum Mean Absolute Error, Percent	11.7	11.2	10.7	10.3	9.7	9.3	9.3	8.6	8.5	8.5

Data Set II

k =	10	20	30	40	50	60
Minimum Mean Percent Error	24.5	23.8	23.0	22.6	22.5	22.8
Minimum Percent of Variance Not Explained	9.4	7.6	6.8	6.2	6.4	6.5
Minimum Mean Absolute Error, Percent	19.1	17.9	17.1	16.5	17.2	16.9

Table 2

While the best fit for each value of k in a wide range has about the same error measures, the values of the multiplier α at which the minimum errors occur vary considerably but systematically as k increases. We will explore this further in Section 4.

3. The Goodness-of-Fit Criterion

Since, in practice, the underlying $f(\underline{x})$ is not known, the various error measures cannot be computed. This brings us to the second question posed in the introduction: How then do we go about selecting σ or α_k and k. (Although we surmise that in actuality we need to estimate only the optimal single parameter value $\lambda = \alpha_k(\overline{d_k})^2/\sigma(d_k)$ in the variable kernel estimates.)

In [2] a goodness-of-fit criterion for a set of samples to a proposed density $f(\underline{x})$ was developed based on the fact that if $f(\underline{x})$ is the true density, then the variables

$$w_j = e^{-nf(\underline{x}_j)V(d_{j,l})}, \quad j=1,\ldots,n$$

where $V(r)$ is the volume of an M-dimensional sphere of radius r, have a univariate distribution that is approximately uniform. Thus, the test statistic for an estimate $\hat{f}(\underline{x})$ is based on the variables

$$\hat{w}_j = e^{-n\hat{f}(\underline{x}_j)V(d_{j,l})}, \quad j=1,\ldots,n .$$

Let $\hat{w}_{(1)} \leq \ldots \leq \hat{w}_{(n)}$ be the ordered permutation of the $\hat{w}_j$. Then the test statistic $\hat{S}$ is defined as

$$\hat{S} = \sum_{1}^{n} (\hat{w}_{(j)} - \frac{j}{n})^2 .$$

One question of great interest to us in this study was whether we could select "good" values of σ or k and α_k by searching for a minimum in $\hat{S}$. The results were affirmative (with one exception we will discuss later).

Naturally, different error measures were generally minimized at different values of the parameters. In Table 3 we list, for every value of k used, the value of α that minimizes each error measure and the value of α that minimizes $\hat{S}$ for that value of k.

For the unimodal case the absolute minimum of $\hat{S}$ occurs at k=100, α=.5. At this point we have

Mean Percent Error = 12.5 (10.8)

Percent of Variance Unexplained = 4.2 (3.6)

Mean Absolute Error, Percent = 8.8 (8.0) .

The figures in parentheses are the minimums of the corresponding measures of error over all ranges and do not occur at a common value of k and α.

In the bimodal case, the minimum of $\hat{S}$ occurred in the original runs at k=60, α=.4. The values at this point were fairly close to the minimums, i.e.,

Mean Percent Error = 22.8 (22.5)

Percent of Variance Unexplained = 10.7 (6.2)

Mean Absolute Error, Percent = 18.8 (16.5) .

For the Parzen Estimator with data set I, the minimizing values of σ for the three error measures above were .40, .35 and .30 respectively. The minimum value of $\hat{S}$ occurred at .60. For data set II, the minimums occurred at .400, .175, .225 and the minimum of $\hat{S}$ at .375. For Parzen estimators $\hat{S}$ indicates "optimal" values of σ considerably higher than the values of σ that minimize the PVNE and the MAE. There is also less consistency between the error measures as to the location of the respective minimizing σ. The σ that minimizes the mean percent error is

Table 3

Minimizing Values of α_k.

DATA SET I

k =	10	20	30	40	50	60	70	80	90	100
Mean Percent Error	1.4	1.0	0.8	0.7	0.6	0.5	0.5	0.4	0.4	0.4
Percent of Variance Unexplained	1.8	1.2	1.0	0.8	0.7	0.6	0.5	0.5	0.5	0.4
Mean Absolute Error, Percent	1.5 or 1.6	1.0	0.9	0.7	0.7	0.6	0.5	0.5	0.4	0.4
$\hat{S}$	1.7	1.2	0.9	0.8	0.7	0.7	0.6	0.6	0.5	0.5

DATA SET II

k =	10	20	30	40	50	60
Mean Percent Error	1.4	0.9	0.7	0.6	0.5	0.4
Percent of Variance Unexplained	1.0	0.6	0.5	0.4	0.3	0.3
Mean Absolute Error, Percent	1.0	0.6	0.5	0.4	0.3	0.3
$\hat{S}$	1.1	0.8	0.6	0.5	0.5	0.4

the highest, and in the bivariate case, considerably higher than the other two minimizing values of σ. Probably this latter fact is due to the behavior of the Parzen estimates at small values of $f(\underline{x})$.

In both data sets, the $\hat{S}$ estimate of σ gives a value of mean percent error close to the minimum attainable for the data set. This is consistently true for the variable kernel estimates also. For each value of k, the $\hat{S}$ minimizing value of α_k has a mean percent error close to the minimum possible for that value of k.

4. Mean Interpoint Distance and the Choice of α

In our various explorations of the variable kernel estimates, we made the empirical discovery that over the range of k investigated, that for both data sets

$$\frac{\alpha_k(\overline{d_k})^2}{\sigma(d_k)} \approx \text{constant}$$

where $\overline{d_k}$ and $\sigma(d_k)$ are the mean and standard deviation of the k^{th} nearest neighbor distances for the data set, and α_k is the "optimal" α for that value of k. To illustrate this, we use as the "optimal" value of α_k, the average of the first three minimizing values given in Table 3. Table 4 gives the values of $\alpha_k(\overline{d_k})^2/\sigma(d_k)$.

The constant decreases about 40% between the two data sets. A similar decrease occurs for those value of σ in the Parzen Estimates which minimize the Mean Absolute Error % and the Percent of Variance Not Explained. It seems clear that the increase in optimal kernal sharpness occurs in order to deal with the increased variability in data set #2.

At the beginning of this study, we used distances to the closest neighbor, next closest neighbor, etc., up to the fifth nearest neighbor. The results were disastrous. Examining the errors, they came mainly from a few points that were too close together. We tried a number of things:

i. Selecting a lower bound D for the interpoint distances and using

$$d'_{j,k} = \max(D, d_{j,k})$$

in the kernel estimate of $d_{j,k}$. D was selected as a percentile (usually either the 5^{th} or 10^{th}) of the $d_{j,k}$, j=1,...,400.

Table 4

$\alpha_k(\bar{d}_k)^2/\sigma(d_k)$

k =	10	20	30	40	50	60	70	80	90	100
Data Set #1	1.3	1.3	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5
Data Set #2	.83	.80	.84	.85	.79	.84	-	-	-	-

ii. Using a weighted average of the first k nearest neighbor distances.

iii. Selecting a multiplicative constant α_k and using $\alpha_k\, d_{j,k}$ or $\alpha_k\, d'_{j,k}$.

None of these helped very much as long as we kept working with k small. The averaging in (ii) was no help. Later we made a theoretical computation in order to find values $\alpha_1,\ldots,\alpha_k$ with

$$\alpha_j \geq 0, \quad i=1,\ldots,k\ , \quad \sum_1^k \alpha_i = 1$$

and such that the variance of

$$\sum_1^k \alpha_i\, d_{j,i}$$

is a minimum. Assuming that the density was "locally constant" so that the distribution of points is "locally Poisson," the answer is

$$\alpha_1 = \alpha_2 = \ldots = \alpha_{k-1} = 0,\ \alpha_k = 1$$

This result gave us some insight into the failure of the averaging process.

In (iii) we found that trying to get more smoothing by increasing α_k led to serious underestimates of the peaks of the densities.

Nothing really helped until we started exploring the larger values of k and found that (iii) worked well when k was large enough.

In terms of what has been empirically learned in this study, we tentatively propose the following method for calibrating a variable kernel density estimate.

Step 1. Pick an initial k equal to some fraction of the sample size, say 10%, or by plotting $\overline{d_k}$ versus k and taking a value of k past the knee of the curve (see figure 9).

Step 2. Do a search for the value of α_k that minimizes $\hat{S}$.

Step 3. Using the minimizing value compute

$$\lambda = \frac{\alpha_k(\overline{d_k})^2}{\sigma(d_k)}$$

Step 4. Vary k in both directions, selecting α_k so as to hold the above ratio constant and search for a k value that minimizes $\hat{S}$.

Note that Step 3 may be dimension dependent.

Figure 5

Constant kernel fit to UNIT NORMAL

$\sigma = .35$

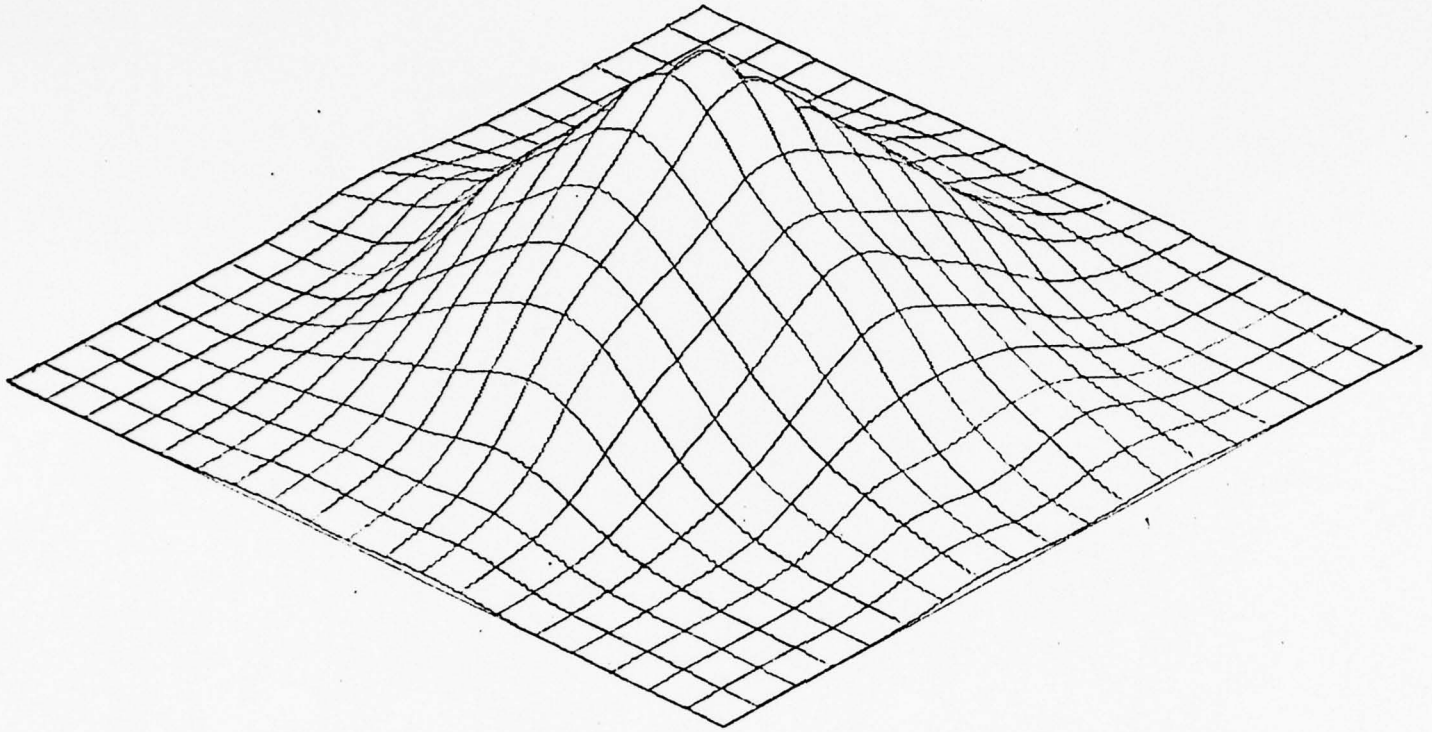


Figure 6

Variable kernel fit to UNIT NORMAL

$k = 60$ $\lambda = .6$

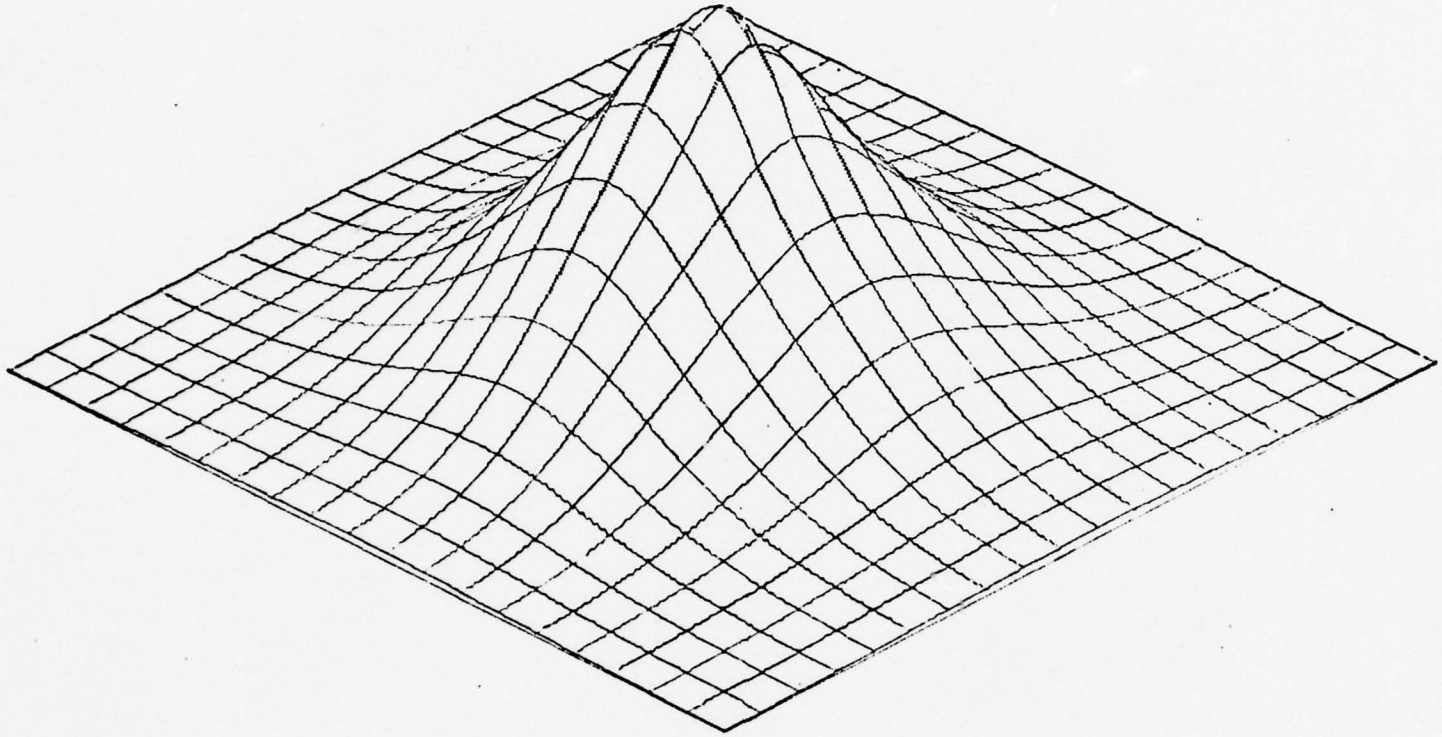


Figure 7

Constant kernel fit to BIMODAL

$\sigma = .275$

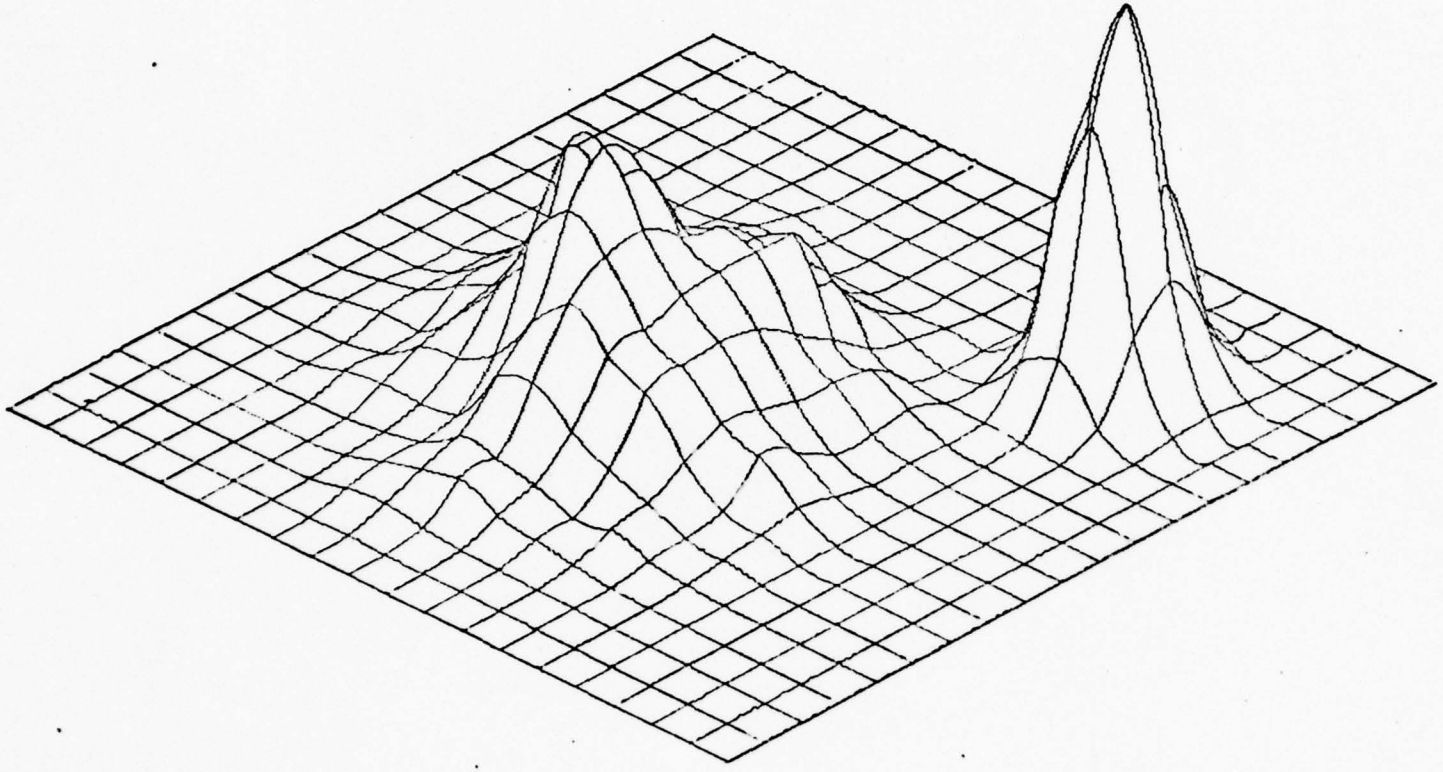


Figure 8

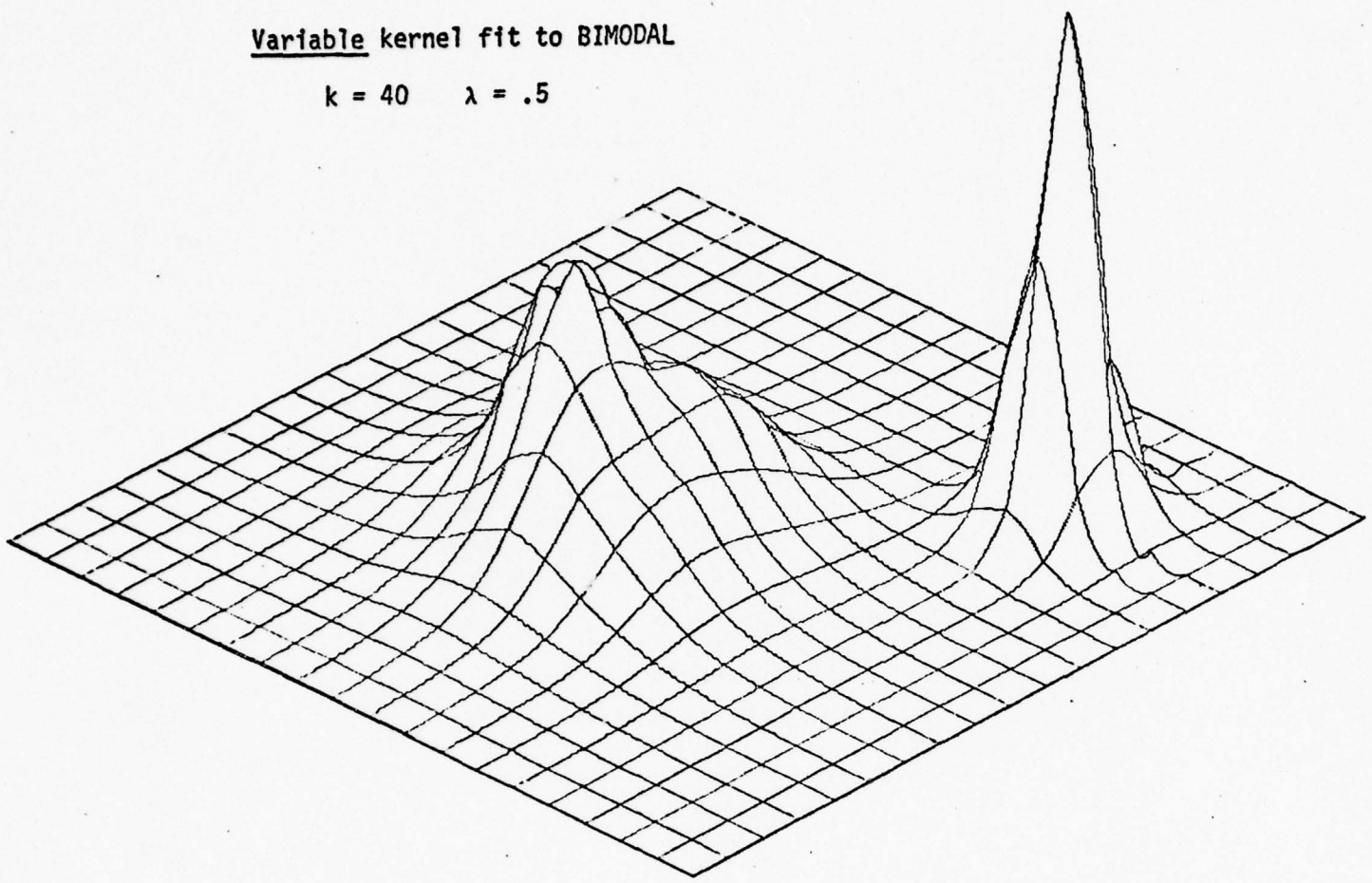
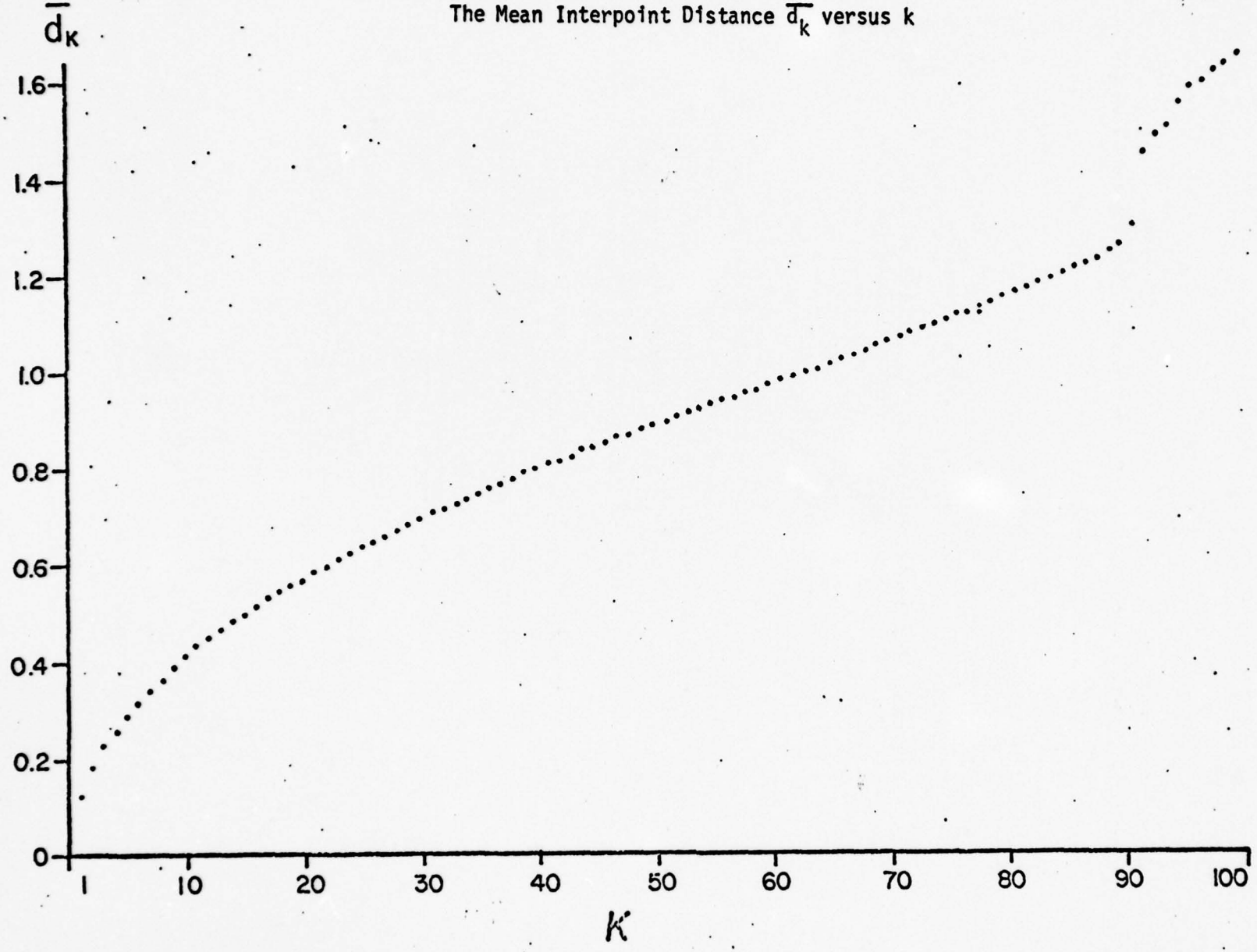


Figure 9

The Mean Interpoint Distance $\overline{d_k}$ versus k



REFERENCES

1. E. Parzen, "On the Estimation of a Probability Density Function and the Mode," Ann. Math. Stat./33, pp. 1065-1076, 1962.

2. L. Breiman, "A General Goodness-of-fit Test for Multidimensional Densities," submitted to JASA, February 1975.

3. L. Kanal, "Patterns in Pattern Recognition," IEEE Trans. on Information Theory, Vol. IT-20, No. 6, pp. 697-722, 1974.

4. E. J. Wegman, "Nonparametric Probability Density Estimation I. A Summary of Available Methods," Technometrics Vol. 11, No. 3, pp. 533-546, 1972.

5. T. M. Cover, "A Hierarchy of Probability Density Function Estimates," Frontiers of Pattern Recognition, Academic Press 1972.

6. P. O. Looftsgaarden and C. P. Quesenberry," A Nonparametric Estimate of a Multivariate Probability Density Function," Ann. Math. Stat., 38, pp. 1049-1051, 1965.

7. J. H. Friedman, J. L. Bentley, R. A. Finkel, "An Algorithm for Finding Best Matches in Logarithmic Time," Submitted to Communications of the ACM, 1974.

8. W. Meisel, "The Complete Pattern Recognition Algorithm," 8th Annual Symposium on the Interface Between Computer Science and Statistics, February 14, 1975, Health Sciences Computing Facility, UCLA.

9. T. J. Wagner, "Nonparametric Estimates of Probability Densities," submitted to IEEE Trans. Information Theory, 1974.

APPENDIX B

TREE STRUCTURED CLASSIFICATION METHODS

Background

Technology Service Corporation has been performing research under a variety of projects involving classification or categorization.

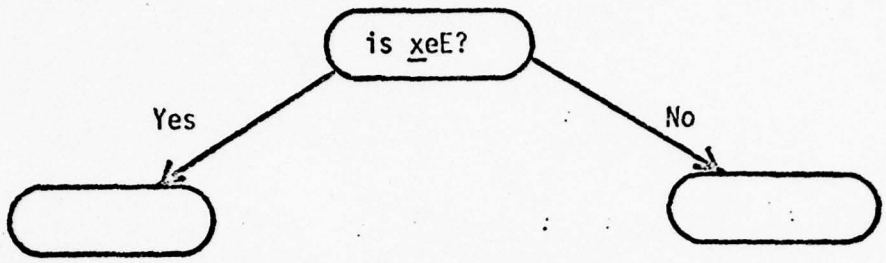
For instance, one project has involved the recognition of ship classes by means of their radar range profiles. In another algorithms were developed for the recognition of spoken words using different speakers. A third project involved the classification of chemical compounds through their mass spectra.

The nature of these problems is such that classical classification techniques, such as the use of Fisher discriminants, etc., are virtually useless. Even recently developed techniques such as the use of nonparametric density estimates or nearest neighbor methods are largely non applicable.

The common elements that make these problems different and difficult is

1. The measurement vector characterizing each object is very high-dimensional.
2. The number of classes is large.
3. The number of classified samples is small, relative to the dimensionality and the number of classes.

To do effective decision making in these problems we have turned to tree structured decision methods. The simplest type of decision tree works this way: denote the measurement vector attached to an object by $\underline{x}$ and let it take values in a space X. A node of the tree corresponds to a subset $E \subset X$. If $\underline{x} \in E$ then the decision is made to pass the object to the left hand node. Otherwise it goes to the right.



The branches of the tree end at terminal nodes. Each terminal node is assigned to one of the classes. When an object passes down the tree and hits a terminal node, it is labeled as being in the assigned class.

This tree structure is in a way fairly simple. It is binary in the sense that each node splits into two descendent nodes, and the decision rules are non-randomized. Still it is the prototype of tree structured classification methods and its successful application in a number of problems has been the stimulus for our recent and proposed research into this area.

To understand why decision trees appear to us to hold great promise in high-dimensional, numerous class problems, think of the simple problem of constructing a word dictionary, containing thousands of entries. In practice, we use a real dictionary very easily and naturally, without realizing what an effective tool it is. First, we look at the first letter of the word. This separates all words into 26 disjoint subsets. Having located which subset we are in, we now look at the 2nd letter of the word. This splits the original subset into 26 2nd generation subsets. We continue until the word is located.

The point is that the decision is not made all at once. The large amount of information carried by the succession of letters in the word is not used at one gulp to decide which one of the thousands of possible cases the word fits into. Instead, a very interesting and practical strategy is used. Starting with a very limited amount of information, namely, the first letter of the word, a rough classification into a small (26) number of groups is done. Then more information (the 2nd letter) is adjoined, and a finer subdivision is made, and so on.

The point is that the high intrinsic complexity of the problem is broken down into a sequence of steps in which highly aggregated information is used to separate a group of objects into a relatively small number of subgroups.

We have emphasized the above, at the risk of being overly simplistic, in order to clarify the simple but powerful idea that underlies decision trees. We know of no other practical method for effectively solving problems characterized by:

- high-dimensionality
- numerous classes
- small sample size.

By aggregating the information and splitting each group into only a few subgroups at each stage, one deals with a sequence of problems having considerably lower dimension and fewer subgroups. Thus, the sample size, relative to dimensionality and number of subgroups becomes much larger.

One very important distinction between classical pattern recognition methods and a decision tree structure is in the way that information is utilized. In the usual pattern recognition approach, the dimensionally is made reasonable by the selection of an apriori mapping from measurement space to "feature" space. That is, depending on certain physical or heuristic principles, the large amounts of detailed information regarding any one object are aggregated and summarized in a small number of variables that comprize the feature vector.

The reason for this mapping is usually very practical: Since the usual pattern recognition algorithms give a one gulp answer, a drastic reduction in dimensionality is necessary both to make the sample size sufficiently dense in the space to define the problem and to make it compatationally feasible.

But having made the reduction in dimensionality, one is stuck with it. The loss in information is irrevocable.

Even if the pattern recognition phase of the problem reveals that additional information would be useful in some regions of the space, it cannot be made available without a reworking of the problem.

The trouble is that the attempt is made to work the problem in two separable non-interacting pieces: one is the feature selection. The second is the classification.

However, in a tree decision structure, there is possible a sequential interaction between classification and information. As one progresses down the branches of the tree, more and more detailed information can be called for by the tree construction algorithm. As a verbal analogy, is

as though at each mode, the tree construction could call upstairs and say "if you want me to do any more separation on these things, then you've got to give me some more information."

However, the problem is not resolved simply by resolving to use a decision tree structure. To a great extent, the problem has been shifted into new ground. It now becomes: how does one determine the most effective or an effective sequence of decision problems defining the tree? That is, what question does one ask at each node of the tree?

Put into a statistical context the problem, simplified to binary trees is this: given that the data vector $\underline{x}$ is drawn with probability p_i from the distribution $P_i(d\underline{x})$ corresponding to the i^{th} class, find a sequence of binary decision questions of the form, "is $\underline{x} \in E$?" that leads to a near maximal probability of correct classification.

In practice, neither the a priori class probabilities p_i or the class distributions $P_i(d\underline{x})$ are known. Instead, one has on hand a set of objects and corresponding data vectors whose classification is known. The hub of the problem is to use these to construct an effective decision tree.

Obviously, by using as many terminal nodes as there are objects in the learning set, we can usually get perfect classification on the learning set. Thus, unrestricted tree growing using the learning set alone will lead to nonsensical results. Restrictions need to be placed on the complexity of the tree relative to the sample size, and some of the already classified objects, chosen in some random way, put aside to act as an evaluation or "test set" for the tree.

Trees share the complex character of all sequential decision methods: a decision made at a point affects all subsequent decisions. Thus, it is

difficult to evaluate what is optimal at each node. With high-dimensional data vectors, one faces an enormous amount of information and at each stage of tree building what is wanted is to use some low dimensional aggregation or "averaging" of the information. But what averaging or aggregation is effective, and how the effectiveness should be measured are difficult questions to resolve.

Thus, there are important problems that need further resolution in order to sharpen the use of decision trees as a practical classification tool. Technology Service Corporation has developed some methods for tree growing and applied them, with very promising results, to problems as diverse as ship classification with the radar range profile as the measurement vector and chemical classification with the mass spectra as the measurement vector.

In the body of this report, we will outline our largely unpublished recent research into decision tree construction and application. Then we will discuss the directions where we believe that further research is needed.

The development of decision tree methodology has important implications for implementation in actual on-line recognition and classification systems. The tree is developed off-line using the given "training set." This is the difficult and time-consuming effort. But the on-line tree consists of a sequence of very simple questions, i.e., a sequence of yes-no questions for a binary tree. Thus, on-line classification can be very rapid. Furthermore, as the more interesting and difficult recognition problems move toward higher dimensions, with more information being extracted concerning each object, tree structuring decision processes become increasingly appropriate and useful.

At the beginning, much of our tree construction was based on heuristics and trial and error, with the resultant misclassification rate being our gauge of success. Over the last year, we have been experimenting with algorithms for the systematic generation of "good" binary trees.

Our best performance to date has been based on the following algorithm: At any node, suppose there are n objects of the learning set with associated measurement vectors $\underline{x}_1,\ldots,\underline{x}_n$. Suppose that of these n objects, n_1 are in class #1, n_2 in class #2,..., and n_J in class #J.

Suppose that we have defined some family $\mathcal{S}$ of potential splits at this node. Each split sends a subset of the n objects to one node, and the complementary subset to the other node. The splits are based on the values of the measurement vectors $\underline{x}_i, i=1,\ldots,n$. That is, each split in $\mathcal{S}$ is based on a question of the form

$$\text{Is } \underline{x} \in E?$$

Hence, in general, the family $\mathcal{S}$ is constructed by selecting a family $\{E_s\}$ of subsets of X and looking at the potential splits generated by

$$\text{Is } \underline{x} \in E_s?$$

At this point, one would like to select the "best" split in $\mathcal{S}$. The problem is how to define "best".

A split at any node impacts all the nodes below it. Therefore, in judging how good a split is, one would theoretically have to trace the subsequent developments of all descendant nodes. A split that does the best possible job in terms of the two immediate descendant nodes might not look very good when the tree is followed down for another generation. Thus, there are levels of judging the goodness-of-split.

Analogously, a novice chess player might, at any given time, choose the move that most improves his immediate position. But a good chess player will think two, three or more moves ahead in considering the implications of the correct move.

To date, we have concentrated on the finding of good criteria for selecting the "best" split from a family $\mathcal{S}$ using only a one generation analysis. This is by no means a trivial matter, because the choice of a good "strategic" criteria can trade-off against a detailed multi-generation analysis. In other words, using the chess game analogy, if a player uses really good criteria for judging his improvement in current position, his criteria will embody a good deal of his past experience gained from learning the future consequences of current moves.

Suppose there was a split that separated all of the objects in class j into one node, and the other classes into the other nodes. For a problem with a large number of classes, this is not a strategically sound split. We would rather get a split such that all the objects in a large subgroup of the classes went into one node and the remaining classes into the other nodes. Therefore, we want criteria for goodness-of-split that rewards the latter type of split more than the former.

The most satisfactory criterion we have found so far is based on the uncertainty measure. For any node N having n_j objects in class j, j=1,...,J define the uncertainty at that node as

$$U(N) = -\sum_j (n_j/n) \log(n_j/n)$$

where $n = \sum_j n_j$. Suppose that a split produces the left N_L and right N_R nodes with n_L of the original n objects going left and $n_R = n-n_L$ going

right. Then define the decrease in uncertainty produced by the split as

$$\Delta U = U(N) - (n_L/n)U(N_L) - (n_R/n)U(N_R)$$

This criterion generally rewards the best strategic split. For instance, if there are J = 2M classes and if a node contains equal numbers of objects in each class, then the splits producing the largest ΔU places all objects in M of the classes in one descendent node and the remaining objects in the other.

The algorithm then searches over all possible splits in S, and selects the one giving the largest ΔU.

The algorithm needs one more piece to be complete. A stopping rule must be specified. Otherwise, as many terminal nodes as there are objects in the test set will be produced. We are currently utilizing the rule: Let N be the original test set population and n, the node population. Set a threshold α and declare the node terminal if there is no split in $\mathcal{S}$ such that

$$\frac{n}{N}\Delta U \geq \alpha \quad .$$

The adoption of this rule and the threshold value of α used were set by heuristics. That is, we generated trees that went down to very small terminal nodes and decided where on the branch it would have been reasonable to stop. The rule was then constructed to more or less match our statistical opinions.

The critical element in this tree growing procedure is the selection of the family $\mathcal{S}$ of potential splits at each node. In ship recognition, the measurement vector consisted of the intensity of radar returns as measured

every two feet along ships ranging up to 500 feet in length. Thus, the measurement vector $\underline{x}$ had a maximum dimensionality of 250. The class $\mathcal{S}$ for all nodes below the 1st generation was generated by questions of the form:

"Does the range profile have a local maximum in the interval [a,b]?"

The ends of the intervals a,b ranged along multiples of 1/100 of the ship's length. Thus a and b were specified by giving two integers L,M $0 \leq L \leq M \leq 100$, and consequently, the split was specified by L and M. Thus, the family $\mathcal{S}$ contained

$$\frac{100 \cdot 99}{2} \simeq 5000$$

potential splits at each node.

In the chemical spectra study, the measurement vectors consisted of peak intensities on a scale of 0 to 100 corresponding to every integer m/e value from 1 up to 320. Thus, $\underline{x}$ was 320 dimensional. The intensities were divided into five logarithmic ranges, sc that the coordinates of $\underline{x}$ could be considered as taking values in the set {1,2,3,4,5}. The family $\mathcal{S}$ was generated by all questions of the form:

"Is the intensity at m/e = k greater than m?"

In other words, each question was characterized by the integers k and m with $1 \leq k \leq 320$ and $1 \leq m \leq 4$. Thus, $\mathcal{S}$ contained about 1200 splits at each node.

New Concepts

The results of these above two studies were exciting, in that we could see that the tree structure gave us a feasible way of solving problems that had previously seemed quite untractable. The algorithm used, for all of its

crudeness and simplicity, produced reasonable classification results. As we worked with it, the drawbacks and deficiencies became apparent, and we could see directions where improvements could produce more powerful and flexible methods. We have outlined some of these in the sections that follow.

Basically, we want to extend and generalize the realm of possible tree structures. One direction we very recently came across is the use of randomized decision rules at the nodes. This leads to structures we have called probability trees and has the promise of resolving two serious shortcomings we have found in practical applications.

Another direction where basic work is needed is in information-adaptive trees. The point here is to allow the class of allowable splits to change as one progresses down the tree, so that near the top, coarse overall features are used and more detailed information is added to discriminate at the lower nodes. Here, possibilities are introduced "looking ahead" to check the consequences of any given split.

Another possibility is that of using other splitting criteria and cleansing terminal nodes have too much of a mixture of classes. Finally, we discuss our thinking in the direction of the "ultimate" decision tree classification method, as applied to a truly numerous class problem.

Boundary Problems, Confidence Statements, and Probability Trees

Analyzing our approach using binary trees and splitting questions of the form

Is $x \underline{\in} E$?

we found that there was, at times, an undesirable sensitivity to the boundary. That is, with many classes to discriminate between, the algorithm would carefully select the splitting set E to include most objects in the test set within a certain set of classes and exclude those without. Often, a considerable number of measurement vectors would fall near the boundary of E. Then when the test set was run, a frequent source of error was due to measurement vectors that just missed being on the right side of the boundary of E at some node and were consequently misclassified.

One reason for this behavior of the boundaries was that our learning set, although large by ordinary one dimensional standards (336 for ships) was sparsely spread out in the high dimensional space. The boundaries could then arrange themselves to do quite well on classifying the learning set and not too well when another sparse set (the test set) was randomly plunked down.

We improved the boundary behavior in ship recognition by taking each learning set profile, adding random noise and in this way generated randomly perturbed profiles from the original profile. But we considered this an artificial baling wire and glue remedy.

The non-randomized character of the decision roles also gave us another difficult problem to solve. When an unknown object goes down the decision

tree and ends in a terminal none labelled as class #J, what confidence we assign to the classification?

To begin with, the terminal node was labelled as class #j if there were more class #j objects in it than any other class. Some terminal nodes have very mixed populations and are terminal because no further discrimination is possible using the class of splits $\mathcal{S}$ and the given stopping role. Other terminal nodes have a very high percentage of their population in one class. This difference in terminal node distribution is complicated by the fact that some unknown objects, in traversing the tree, come down through nodes at which their measurement vectors are very close to the boundary. Others stay away from the boundary at all nodes.

The problem did not seem to have a really satisfactory formulation of binary decision trees. Although we conjured up some measures of confidence, we had little confidence in them.

In July 1976 we have started thinking about a different type of tree structure that may solve both the boundary problem and the problem of assigning confidence to the classification of an unknown object.

These new structures are called probability trees and are described as follows:

Start with a learning set consisting of n objects in J classes, $C_1,\ldots,C_J$. Number the objects $1,\ldots,n$. A node of the tree corresponds to a vector $\bar{p}$ of probabilities $(p_1,\ldots,p_n)$. At each node, define

$$\bar{p} = \sum_i p_i$$

and the class probabilities P_j by

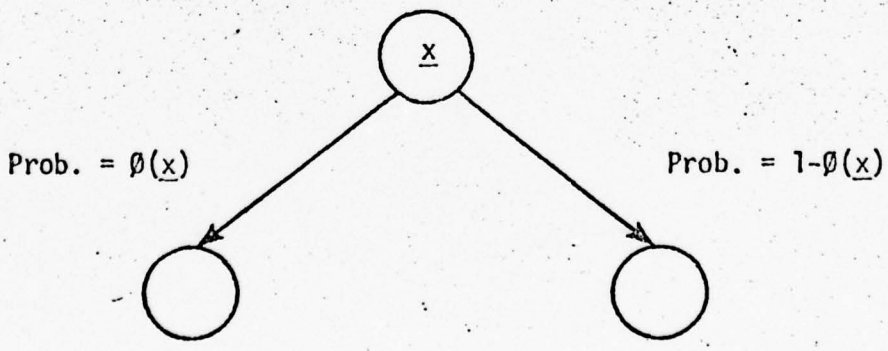
$$P_j = \sum_{i \in C_j} p_j / |\overline{p}|$$

where $i \in C_j$ denotes the sum over all objects in class j.

We will set up a procedure for splitting this node. Instead of considering a set E and asking: is $\underline{x}$ E?, define a family $\mathcal{F}$ of functions $\phi(\underline{x})$ on X, such that each $\phi(\underline{x})$ satisfies

$$0 \leq \phi(\underline{x}) \leq 1$$

We will use a particular function ϕ as follows: for an object having measurement vector $\underline{x}$, put it into the left node with probability $\phi(\underline{x})$ and the right node with probability $1-\phi(\underline{x})$. Thus we get the picture



If the i^{th} object has measurement vector $\underline{x}_i$, and its probability in the present node is p_i, then its probability in the left node is

$$p_i(L) = \phi(x_i)p_i$$

and in the right node is

$$p_i(R) = (1-\phi(x_i))p_i \quad .$$

Thus we get the left and right node probability vectors

$$\overline{p}(L) = (p_1(L),\ldots,p_n(L))$$

$$\overline{p}(R) = (p_1(R),\ldots,p_n(R))$$

Note that

$$\overline{p}(L) + \overline{p}(R) = \overline{p}$$

Also, the class probabilities $P_j(L)$ and $P_j(R)$ are defined as before, i.e.,

$$P_j(L) = \frac{1}{|\overline{p}(L)|} \sum_{i \in C_j} p_j(L)$$

Defining uncertainty in a set of class probabilities

$$U(P) = -\sum_j P_j \log P_j$$

we have that the decrease in uncertainty due to the node split is

$$\Delta U = U(P_j) - \frac{|\bar{p}(L)|}{|\bar{p}|} U(P(L)) - \frac{|\bar{p}(R)|}{|\bar{p}|} U(P(R))$$

Choose the ϕ in $\mathcal{F}$ that maximizes ΔU (see remarks later).

Continuing this way, we arrive at M terminal nodes $T_1,\ldots,T_M$ with corresponding vectors of probabilities $\bar{p}_1,\ldots,\bar{p}_M$ and with the m^{th} node having class probabilities $P_j^{(m)}$, $j=1,\ldots,J$.

Given a test object, we get our results as follows: traversing the tree and using the selected function $\phi(\underline{x})$ at each node, we end up with probabilities q_m, $m=1,\ldots,M$ that the object is in terminal node T_m. Define the probability that the object is in class j by

$$P(\text{object is in } C_j) = \sum_m P_j^{(m)} q_m$$

Identify the object as being in that class having the largest probability. Then the probability of misclassification is

$$1 - \max_j P(\text{object is in } C_j)$$

A few remarks: The standard way of node division, i.e., asking the question is $\underline{x} \in E$? can be formulated as; let

$$\phi(\underline{x}) = \begin{cases} 1, & \text{if } \underline{x} \in E \\ 0, & \text{otherwise} \end{cases}$$

and sending the object left with probability $\phi(\underline{x})$, right with probability $1-\phi(\underline{x})$. Thus, using a $\phi(\underline{x})$, subject to $0 \leq \phi(\underline{x}) \leq 1$ is a generalization of the above.

Now, it is impossible and undesirable to maximize ΔU over too large a class of functions $\phi(\underline{x})$. The reasons are:

1. One does not want to allow too complicated a set of decision rules.

2. If the class of allowable functions $\phi(\underline{x})$ is too large, the minimization search will take too long.

Thus, the essential ingredient for this method to work is the selection of an appropriate family $\mathcal{F}$ of functions to maximize ΔU over. Probably reasonable requirements are

- All of the functions ϕ in $\mathcal{F}$ be reasonably smooth.
- The family $\mathcal{F}$ depends only on a small number of parameters.

Note that, in a sense, a probability tree makes use of fuzzy decision sets E. If $\underline{x}$ is well within or outside of E it is sent to one or the other node with probability close to zero on one. But if it is close to the boundary the decision becomes blurry.

Interestingly enough, although the impedus for probability trees was to get rid of the boundary problem, as an unforeseen by-product we got a natural solution to the confidence problem as expressed by the probability that the unknown object was misclassified.

Information-Adaptive Trees

A decision tree structure that uses the same set of potential splits at every node has some serious disadvantages. It is generally using too much information during the early part of the tree construction and not enough during the later part.

For instance, in the chemical spectra data base, we used the 1600-dimensional feature vector consisting of all peaks at m/e locations between

1 and 320 and a separation of the intensities into five ranges. That gave the 1280 potential splits: Is the intensity (coded) of the peak at m/e = k greater than j, j=1,2,3,4 and k = 1,2,...,320. In general, only a small fraction of the spectra in the data base had a peak of any given m/e value whose intensity was greater than j,j=1,...,4.

Thus, at the early stages of the tree we had too much splintering with small nodes being split off. In the later part of the tree construction, the spectra at any node have been filtered down by their common responses to a number of questions. They exhibit more and more similarity the lower down in the tree the node is. Therefore, to effectively split a small node, we may need to include more detailed information about the spectra, or to ask more detailed questions about it.

In both of our studies, many of the low-confidence small nodes cou'd not be effectively split by one of the available splitting questions. This may have been due to the fact that the set of potential splits was not able to get at the level of detail needed further down the tree.

Another possible cause for the program's difficulty in separating some small nodes is its restriction to a single stage optimization procedure. It always selects the split which gives the greatest decrease in uncertainty. This is a "one-step optimal procedure." But usually, the succession of two "one-step" optimal splits will not be the "two-step" optimal split. Instead of judging the merit of a split on how much the split reduces the uncertainty, we could use the following procedure: For each allowable split of the node in question into two nodes, find

the optimal split of each of the two nodes. Now compute the decrease in uncertainty as we go from the original node to the four "descendant" nodes and use this decrease to judge the merit of the original split. This gives a "one-step look ahead" evaluation of the split. As an analogy, this corresponds in a chess game to a chess player who is capable of looking ahead one move into the future as contrasted to a chess player that plays the move that yields him the most immediate gain.

The point is that, without a look-ahead potential, especially in the later stages, we may be selecting "blind alley" splits; that is, splits that look good as immediate prospects, but that do not eventually lead to high-confidence nodes.

We believe that a substantial improvement in decision tree classification procedures can be made by using an information-adaptive structure. By this we mean a decision tree that

a. in the early stages uses less detail, aggregates the information, and splits the objects into a few large general categories

b. in the later stages, adds more detailed information, examines a larger class of splits, and, if necessary, goes into a look-ahead mode of operation.

This type of tree structure also lends itself to greater efficiency in construction. At the early stages, with a large data base, each node contains many items. With the chemical spectra base, the first few nodes contained thousands of compounds. If one used the same degree of detail

at this stage that is desirable later, then the number of potential splits will number in the thousands. The search through all potential splits over a large number of items is very time-consuming and the rewards at this level are not commensurate with the effort expended.

When the nodes are small, then the search over a larger number of allowable splits is not as burdensome computationally.

One way an information-adaptive structure could be implemented is as follows: A number of different classes $\mathcal{S}_1,\ldots,\mathcal{S}_J$ of splitting questions will be defined. At the top level is a relatively small class $\mathcal{S}_1$ of questions that uses highly aggregated information about the objects. At the bottom level is a large class $\mathcal{S}_J$ of splitting questions that uses the most detailed information about the objects. The tree growing will proceed as follows:

1. At any node, depending on its size relative to the original total population, fix a threshold value T(s), start with class $\mathcal{S}_i$ where $\mathcal{S}_i$ is determined by T(s), and find the best split generated by this class.
2. If the decrease in uncertainty associated with the best split in $\mathcal{S}_i$ is not greater than the threshold value T(s), then examine, in a similar way, all the allowable splits in $\mathcal{S}_{i+1}$.
3. Continue in this way until $\mathcal{S}_J$ is exhausted. If no allowable split exceeds the threshold value, then
   i. either call the node terminal,

or

ii. go into a one-step look-ahead decision mode.

Which alternative we choose in 3 above will depend on the size and confidence level of the node in question.

The usefulness of highly aggregated information at the beginning of the decision structure is vividly illustrated in chemical spectra study. If we had used the set of 320 splitting question, "Is the molecular weight equal to k, k=1,2,...,320?," none of these splits would have caused an appreciable decrease in uncertainty. The reason is that only a small proportion of the compounds have a given molecular weight. The definition of uncertainty is such that when a node is split and one of the two descendent nodes has a very small fraction of the parent's population, then the decrease in uncertainty is small. Thus, if we had used the full detail available in the molecular weight information, the probability is that it would have caused very little, if any, alteration in the tree structure and in the misclassification rate.

However, aggregating the molecular weight information by dividing it into the two subsets--even weights and odd weights--and adding only the single question as to which subset the weight was in, led to a different initial split and a drastic reduction in misclassification rate.

In our thinking about the chemical spectra problem, we could see the following possibilities of different levels of aggregation.

The first level of questions could include questions of the form: Are there two or more main peaks in the M/e sequence k,k+14,k+28,k+42,...

for k an integer between 0 and 13? Or, more generally, one could ask questions of the form: Are there j or more main peaks in the subset of m/e values $(C_1, C_2, \ldots, C_N)$?

Notice that on this level we would be ignoring the intensities of the peaks and using only their locations.

At the next level $\mathcal{L}_2$ of questions, the intensities might be introduced. For instance, a possible set of questions at this level might be: Are there j peaks with intensities i in the subset of m/e values $\{C_1, C_2, \ldots, C_N\}$? Going down the levels, the questions become more detailed. For instance, we may want to go to the level of questioning of the form: Is there a peak at m/e ε C_1 with intensity i_1 and a peak at m/e ε C_2 with intensity i_2?

At a certain level down the tree, the isotopic information should be added to the main peak information. With this added information, questions concerning the ratio of intensities of main peaks and nearby isotope peaks may aid in the separation of classes at the nodes.

A one- or multiple-step look ahead procedure is costly in its initial construction. For example, if there are N splitting questions at the level being used, then for a one-step look-ahead optimization, $2N^2$ splits have to be examined. This number comes from observing that for such allowable split of the original node, all allowable splits of the two descendant nodes are examined. If there are, as in our study, about 1000 allowable splits, then a one-step look-ahead optimization would have to examine $2\ 10^6$ splits. Hence, the one-step look-ahead procedure would have to be planned in a modified form in order to be feasible.

Other Splitting Criteria and Iterative Cleansing

As we mentioned before, the uncertainty has the property that if a split results in one node much smaller than the other, then there is not much decrease in uncertainty. This may be undesirable in some situations. For example, suppose there are 5000 objects in a node, and a certain split results in a node containing 100 objects and another with the 4900 remaining objects. Even if the node with 100 is absolutely pure, that is, consists entirely of one class, the split will not produce much of a reduction in uncertainty and will almost surely not be the optimal choice.

It is possible that we want to choose our criteria for goodness-of-split so that it gives a higher weight to splits that produce fairly pure nodes that are above some minimal size.

For instance, at one extreme, we could adopt this strategy: Find the split that produces the largest node having more than 90 percent purity (i.e., such that more than 90 percent of its population is in one class). Having split off this node, repeat the procedure on the other node until no 90-percent pure node greater than some minimum size can be found. Now that we have chipped away all the above 90-percent pure nodes, we lower our level to 80 percent and search for the largest node having at least 80 percent purity. Having found an 80-percent node, we then try to extract an above 90 percent chip off of it by splitting.

This chipping away of small pure nodes procedure does not seem desirable near the early stages of tree structure. At this initial phase, the best thing to do is to get the population roughly sorted into large

subsets such that each contains significant numbers of one or more classes, but excludes most of the members of one or more of the remaining classes. At the advanced levels, when we try to split nodes containing only two or three classes, the "pure chip" strategy itself or in combination with the uncertainty criteria may produce more accurate classification.

Thus, we may want to adapt our splitting criteria to the depth of the part of the tree we are working on.

There are an infinity of other criteria for judging goodness-of-split. But in a sense, the "pure chip" and reduction of uncertainty criteria stand at opposite ends of this continuum of possibilities.

Even with improved tree-growing procedures, it is inevitable that some of the terminal nodes will have low-confidence levels, where the confidence level is defined as the percentage of the largest class in the total population of the node. A simple iterative procedure for cleansing the low-confidence terminal nodes is to pool the populations of all terminal nodes with a confidence level below a pre-set level (say, for instance, 80 percent) and <u>using this pooled population as the initial population</u>, rerun the tree construction program. Now that the fairly pure terminal node populations have been pulled out, the tree structure for classifying the remaining population should be entirely different than the original tree.

This iteration can, of course, be repeated over and over, but we strongly suspect that the point of diminishing returns will appear after one or two iterations. Still, we anticipate that a substantial improvement can be produced by this iterative cleansing.

Further Moves Toward a Numerous Class High Dimensional Tree Classifier

The problem we want to ultimately be able to solve is something like spoken word recognition with a large possible vocabulary, say 1000 words. We want to be able to recognize an individual word, no matter who speaks it. Now attached to each spoken word is the high dimensional measurement vector consisting of the digitized recording of the word.

Our planned strategy, in attacking this problem, will be to first use highly aggregated information to separate the words into a few subgroups. For instance; is it a multisyllable word? Does a peak appear near the beginning of the word in this frequency range?

The tree structure will be information adaptive, at each node, the subgroup of word classes present will be broken down in smaller subgroups.

Obviously, this is a sensible approach. The question is how to implement it? We will describe a general framework and then look at some specific examples.

Step I: Select a family $\{T_\theta\}$, $\theta \in \Theta$, which maps the measurement vectors of the learning set into a low dimensional space (feature space).

Step II: Using the values $\underline{y} = T_\theta(\underline{x})$ corresponding to the feature vectors of the job class, estimate the density $f_y(\underline{y})$. If the apriori probabilities of class J are p_j then the probability that an unknown item with feature map $\underline{y}$ belongs to class J is defined to be

$$P(j|y^*) = \frac{p_j f_j(y^*)}{\sum_j p_j f_j(y^*)}$$

Step III. For all items in the i^{th} class, let C_i be the set of feature vectors. Define

$$n_{ij} = \sum_{\underline{y} \in C_i} P(j|\underline{y})$$

Step IV. Take the "confusion" matrix $||n_{ij}||$ and cluster the classes into similar groups. Define a measure of misclassification between groups. After the clustering, let this measure be $M(\theta)$.

Step V. Select θ to minimize $M(\theta)$.

Step VI. Repeat the above 5 steps on each of the subgroup of classes defined in the 4th & 5th steps above, using another family of feature maps.

This general framework gives rise to a decision tree structure, although it is not necessarily binary. It can be made binary by clustering the classes into two groups at each stage, but this is unnecessarily refrictive. A better alternative that a binary grouping on any fixed number of dissimilar groups is to allow the existence of a group of "dubious" classes. So, for example, a good ternary clustering world consists of two extremely dissimilar groups of classes and a "dubious" group between them.

This approach is a combination of a number of methodologies, including clustering. One problem we face in implementing an algorithm of the above type is to find an effective way of clustering the classes into subgroups at each stage. We have developed one such algorithm which is described in Appendix D.

Other Related Research

In his extensive 1974 review [1] of pattern recognition methods, Laveen Kanal states "For the multiclass case, most of the work has either cast the M-class problem as M(M-1)/2 two-class problems or employed multidimensional scatter ratios popular in classical multiple discriminant analysis." Thus, the use of tree structured decision methods in multiclass problems (M large) is relatively novel in the area of classification or pattern recognition. However, there is some scattered work in the literature that is relevant to our proposed research. One of the earliest works in the field proposing the use of tree structured classification is due to W. Meisel [2], the proposed co-principal investigator. Some related but very specialized results regard "tree grammars." These are referenced in Kanal's work [1].

We are certainly not the first to propose the use of the uncertainty measure as a splitting criteria, and the idea has appeared sporadically [3,4]. Tree structured decision methods appear very prominently in clustering theory. Hartigan's recent and excellent survey book [5], contains a full discussion and description of their use.

One of the most successful use of tree structured decision methods is the nonlinear regression program AID and its successors developed at the University of Michigan and widely used in the social sciences, [4,6]. In this algorithm a search is made for successive splits over the independent variable space. The splitting criterion is the reduction in mean square error in predicting the dependent variable. The result is a tree structure of binary splits of the data points.

Another use of decision tree structures is in file search procedures. A good account appears in [7].

BIBLIOGRAPHY

1. Kanal, L., "Patterns in Pattern Recognition: 1968-1974." IEEE Transactions on Information Theory, Vol. IT-20, No. 6, Nov. 1974, pp 697-722.

2. Meisel, W. S. and D.A. Michalopoulos, "A Partitioning Algorithm with Application in Pattern Recognition and the Optimization of Decision Trees," IEEE Transactions on Computers, Vol. C-22, Jan. 1973, pp 93-103.

3. Ritter, G. L., S. R. Lowry, H. B. Woodruff, and T. L. Isenhour, "Relationship between Mutual Information and Classification," Analytical Chemistry, Vol. 48, No. 7, June 1976, pp 1027-1031.

4. Mandel, L. and R. Messenger, "A Modal Search Technique for Predictive Nominal Scale Multivariate Analysis," Journal of the American Statistical Association, Vol. 67, Applications Section, pp 768-772.

5. Hartigan, J., Clustering Algorithms, (1975), John Wiley and Sons, New York, N.Y.

6. Sonquist, J. A., Multivariate Model Building: The Validation of a Search Procedure (1971), Institute for Social Research, University of Michigan, Ann Arbor, Michigan.

7. Bentley, J. M., "Multidimensional of Binary Search Trees used for Associative Searching," Communications of the ACM, Sept. 1975, Vol. 18, No. 9, pp 509-517.

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: AFOSR - TR - 77 - 0376	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): TOPICS IN THE ANALYSIS AND OPTIMIZATION OF COMPLEX SYSTEMS	5. TYPE OF REPORT & PERIOD COVERED: Final rept.
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): William S. Meisel, Leo Breiman	8. CONTRACT OR GRANT NUMBER(s): F44620-76-C-0069
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Technology Service Corporation, 2811 Wilshire Blvd, Santa Monica, CA 90403	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 61102F / 2304/A6
11. CONTROLLING OFFICE NAME AND ADDRESS: Air Force Office of Scientific Research/NM, Bolling AFB, Washington, DC 20332	12. REPORT DATE: 28 Feb 77
	13. NUMBER OF PAGES: 29
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): 62p.	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This one-year contract was in essence a continuation of a previous five year effort. The purpose was to discover more effective methods of analyzing high dimensional data sets. The motivation was the conviction that classical methods were often inappropriate and led to misleading results. A technical paper entitled "Variable Kernel Estimates of Multivariate Densities and Their Calibration" was accepted for publication in Technometrics. The problem treated was to estimate some unknown probability density in m variables based on n →
next page

20. Abstract

cont

independent samples from that distribution. The solution is an extension of a method proposed by Parzen. Additional work was concerned with completing two other papers that were almost finished during the previous year in the areas of goodness-of-fit and classification methods.